IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	G. Paul Koning, Peter C. Hayden and Paula Long		
Application No.:	10/762,984	Group:	2188
Filed:	January 21, 2004	Examiner:	Doan, Duc T.
Confirmation No.:	5998		
For:	Block Data Migration		

AMENDED APPEAL BRIEF

Mail Stop Appeal Brief Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

This Amended Appeal Brief is submitted pursuant to the Notification of Non Compliant Appeal Brief mailed from the U.S. Patent and Trademark Office on August 26, 2009, further pursuant to the Notice of Appeal received in the U.S. Patent and Trademark Office on February 12, 2009, and in support of the appeal from the final rejection(s) set forth in the Office Action mailed on September 10, 2008. The fee for filing a brief in support of an appeal has already been submitted.

I. REAL PARTY IN INTEREST

The real party in interest is EqualLogic, Inc., 300 Innovative Way, Suite 301, Nashua, NH 03062. EqualLogic, Inc. is the Assignee of the entire right, title and interest in the subject application, by virtue of an Assignment recorded on January 21, 2004 at Reel 014930, Frames 0827-0830.

II. RELATED APPEALS AND INTERFERENCES

Appellants, the undersigned Attorney, and the Assignee are not aware of any related appeals or interferences which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

III. STATUS OF CLAIMS

Claims 1 - 10, 12, 14 - 21 and 23 have been finally rejected. Claims 11, 13, 22 and 24 were previously cancelled. All remaining Claims 1-10, 12, 14-21 and 23 are being appealed herein. A copy of all remaining claims appears in the Appendix of this Brief, with the status of each claim (e.g., rejected, allowed, withdrawn, objected to) indicated therein.

IV. STATUS OF AMENDMENTS

No amendments are being made concurrently with this Appeal Brief. No amendments have been filed subsequent to the Office Action made Final dated September 10, 2008.

V. SUMMARY OF CLAIMED SUBJECT MATTER

Claim 1

Claim 1 is directed to an apparatus for data storage resource migration. The apparatus comprises a plurality of storage servers (see Fig. 8, "storage servers" 161, 162, 163 and the text at page 15, line 33 to page 16, line 5).

Each server has a set of resources stored thereon (Fig. 8, items 280 and the text at page 16, line 9-11, "…for the purposes of illustration only, FIGURE 8 represents the resources as pages of data 280").

The resources are partitioned among the servers (Fig. 9 and page 16, lines 24-28, "In the routing table embodiment depicted in FIGURE 9, the equivalent servers support two partitioned volumes. A first one of the volumes (volume 18) is distributed or partitioned across all three equivalent servers 161, 162 and 163. The second partitioned volume (volume 17) is partitioned across two of the equivalent servers, servers 162 and 163").

The storage servers each also have a load monitor process (See Fig. 8, blocks 220A, 220B, 220C, "load monitor" and the text at page 16, line 29 to page 17, line 1) capable of communicating with other load monitor processes for generating a measure of loading on respective ones of the plurality of servers (See Fig. 8 and the text at page 17, lines 1-10, "...the load monitor processes are capable of communicating among each other").

The apparatus also transfers resources from one of said plurality of servers to another server in response to the measure of loading. (See Fig. 8 and page 17, lines 23-24, "the load monitor and resource allocation processes conclude that page 280 should be moved to server 162".)

A process (Fig. 8, resource transfer process 240A, 240B, 240C) detects when a resource write request is being made to a portion of a partitioned resource that is in a process of being moved from the first server to a partitioned second server (Fig. 8, page 13, lines 1-9, "According to one embodiment, a page to be migrated is considered to be "owned" by the originating server...Requests to write new data into the target page are handled specially: data is written to both the page location at the originating server and to the new (copy) page location at the destination server... In one embodiment, it is the resource transfer process 240 depicted in FIGURE 8 that carries out this operation").

In response to a write failure on the second server the migration process for that resource is restarted (Fig. 8, page 13, lines 15-17, "Such write processing approaches are necessary to support the actions required if a failure should occur during the move, such as a power outage. If the page is moved as a single unit, an aborted (failed) write can begin over again from the beginning").

Support for Claim 1 can also be found elsewhere in the application as originally filed such as at least at page 4, line 26 through page 5, line 17; page 12, line 25 through page 15, line 5; Tables 1 and 2; and Fig. 8 and 9.

Claim 6

Claim 6 depends from Claim 1 and adds a requirement that the system determine whether a server is servicing a disproportionate share of client requests.

Support for this claim is found at least at page 17, lines 21-23; Fig. 8, and elsewhere.

Claim 8

Claim 8 also depends from Claim 1 and adds that a routing table is used for tracking resources maintained on the system.

Support for this claim is found at least at Fig. 8, items 200A, 200B, 200C and page 16, lines 16 to 19; and elsewhere.

Claim 14

Independent Claim 14 is directed to a process for moving resources (See Fig. 8, items 280 and the text at page 16, line 9-11, "...for the purposes of illustration only, the FIGURE 8 represents the resources as pages of data 280") across a plurality of storage servers (Fig. 8, "storage servers" 161, 162, 163, and the text at page 15, line 33 to page 16, line 5). The servers each have a set of resources partitioned thereon (See Fig. 9 and page 16, lines 24-28, "In the routing table embodiment depicted in FIGURE 9, the equivalent servers support two partitioned volumes. A first one of the volumes (volume 18) is distributed or partitioned across all three equivalent servers 161, 162 and 163. The second partitioned volume (volume 17) is partitioned across two of the equivalent servers, servers 162 and 163").

The process also comprises a step OF monitoring a server load (See Fig. 8, blocks 220A, 220B, 220C, "load monitor" and the text at page 16, line 29 to page 17, line 1), and communicating with other load monitor processes for generating a measure of loading (See Fig. 8 and the text at page 17, lines 1-10, "...the load monitor processes are capable of communicating among each other").

The process also transfers, as function of the measured load, a resource from one server to another server (See Fig. 8 and page 17, lines 23-24, "the load monitor and resource allocation processes conclude that page 280 should be moved to server 162").

The process also detects when a resource write request applies to a resource that is in the process of being moved or "migrated" from one server to another (Fig. 8, resource transfer processes 240A, 240B, 240C, and page 13, lines 1-9, "Requests to write new data into the target page are handled specially: data is written to both the page location at the originating server and to the new (copy) page location at the destination server... In one embodiment, it is the resource transfer process that carries out this operation").

The migration process is restarted from the beginning upon a failure (Fig. 8, page 15, lines 15-17, "Such write processing approaches are necessary to support the actions required if a failure should occur during the move, such as a power outage, if the page is moved as a single unit, an aborted (failed) write can begin over again from the beginning").

Support for this claim is also found at least at page 4, line 26 through page 5, line 17; page 12, line 25 through page 15, line 5; Tables 1 and 2; and Fig. 8.

Claim 17

Claim 17 depends from claim 14 and includes a further step of determining whether a server is servicing a disproportionate share of client requests.

Support for this claim is found at least at page 17, lines 21-23; Figure 8 and elsewhere.

Claim 19

Claim 19 depends from claim 14 and includes a further step of maintaining a routing table for tracking resources maintained by the system.

Support for this claim is found at least at Fig. 8, items 200A, 200B, 200C, page 16, lines 12 - 19; and elsewhere.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Whether Claims 1-10, 12, 14-21 and 23 are properly rejected under 35 U.S.C. 103(a) as being obvious over Blumenau, *et al.* (U.S. Publication 2004/0080558).

Whether Claim 5 is properly rejected is under 35 U.S.C. 103(a) in view of Blumenau and Applicants' Admitted Prior Art (APA).

VII. ARGUMENT

There are several reasons why Blumenau fails to teach aspects of Applicants' claims and thus cannot render any of the claims obvious.

The Applicants' Claimed Invention

The Applicants' invention is directed to approaches for controlling a server-based, partitioned resource, storage system. In such a system, a plurality of storage servers store a set of partitioned resources. The resources are "partitioned", meaning that portions of a given resource are stored on one server, and other portions of that very same resource are stored on other servers. The storage servers each have a respective load monitor processes that communicates with the load monitor processes in the other servers to determine a measure of loading at each respective storage server. Based on this determined measure of loading, portions of the partitioned resources are transferred back and forth, from one server to another, in a so-called "migration". A write detect process detects when a resource is in the process of being moved ("migrated") between a source server (the "first server") and a second server (the "target server").

The claims are specifically directed to what happens when a write failure occurs on the target server (i.e., the "second server" as recited in the claims) during migration. This is handled in a simple and efficient way – by restarting the migration process for the portion of the resource which experienced the failure.

Accordingly, Claim 1 recites in pertinent part:

> a resource migration process for transferring a resource …
> in response to said measure of loading;
> a write-detect process which:
> (i) detects when a resource write request applies to a resource that is in the process of being moved from a first server to a second server …; and
> (ii) in response to a write failure on the second server, restarts the migration process.

1. *Blumenau does not detect target server write failures in a resource migration process, and in response thereto, then restart that very same migration process.*

Blumenau discloses a write error recovery process, and to that extent, is similar to Applicants' claimed invention. Blumenau specifically compares source and target data with state information to determine which data is still "good" and which data is "bad" (see paragraph [0048], lines 2-5 and paragraph [0049], lines 2-7). For example, the state information may be a count which indicates the number of data base operations performed on a particular storage location (see paragraph [0051]). Blumenau's recovery process then specifically copies the good data from the storage location where the write completed successfully to the other location, based on the state information. See paragraph [0048], lines 5-7 and [85], lines 9-12. Alternatively, Blumenau's recovery process can invalidate data stored at both locations. See paragraph [0048], lines 20-23 and paragraph [85], lines 9-11.

The Examiner points to Blumenau at paragraphs [0039 through 0040] as supposedly teaching the features of almost all of Applicants' invention. The Examiner furthermore believes that Blumenau's paragraph [0042] discloses that when the write is not completed, "a migration process is restarted", pointing to Blumenau's specific statement that the "DBMS will reissue the request".

However, the mere suggestion that a Data Base Management System (DBMS) reissue a write request for a particular piece of data is not the same thing as restarting a resource migration process. Applicants' claimed resource migration is the process of moving the entire partitioned resource (e.g., a large chunk of data) to a different location. The claimed invention requires attempting to move the entire resource once again.

Blumenau is also suggesting only that a DBMS reissue a specific failed write request at the DBMS level. Applicants' restart of a migration process is transparent at the DBMS or other application level. It is not a response to an application level, failed request to access a specific database record.

In other words, Blumenau teaches only that a high level application (i.e., a database management system) can reissue a write request when that write operation has not been completed within a certain amount of time. Applicants, on the other hand, restart a "resource migration process" that is responsible "for transferring a resource" from one server to another "in

response to said method of loading" (i.e., at a different level which would be invisible to an application level DBMS process).

Furthermore, according to paragraph [0048] of Blumenau, undesirable states may be eliminated by a recovery process that determines whether or not the source and target data is consistent, and when it is not, takes action to make the data consistent. For instance, to recover from an interrupted migration, a comparison can be made between the state information for corresponding data storage locations in the source and target volumes. If, after the comparison, it is determined that a more recent write operation was performed to either of the source or target data storage locations, that data stored in that particular data storage location may be relied upon as good data.

Again, Blumenau is clearly only suggesting a much more involved process of checking to see whether each data write operation was actually successful or not. This, again, is not the same thing as, or suggestive of, restarting an entire migration process.

With respect to other aspects of the Examiner's argument, we note that Blumenau also does not actually disclose that the server processes communicate with each other to generate a measure of loading on the respective servers as claimed. The Examiner points to Blumenau's state table 105 as providing this information. However, that table stores state information concerning backup, copy, and recovery. It does not include, suggest or teach maintaining a data indicative of load across a plurality of servers or doing anything in response to the same, never mind being used to restart a migration process, as claimed.

2. *Blumenau teaches away as he requires keeping state information.*

Applicants also note that Blumenau requires that write state consistency information be maintained. Indeed, Blumenau specifically states his is a technique for recovering the state of write process without having to reperform operations that were completed successfully before the interruption (see the Abstract and paragraph [0029]). In contrast to this, Applicants' invention does not maintain state information, and specifically requires restarting the migration process.

Blumenau uses a state information table 105 that stores the state of pending I/O operations. As explained in paragraph [0034], the state information table 105 provides an indication of the status of input/output (I/O) operations performed on one or more storage

locations of storage system 102. Blumenau then goes on to explain that when a write is performed on one or more portions of data 203 on volume 201, an indicator is updated in the state change information table (Blumenau, paragraph [0034]). This information is a bit that indicates that a change has been made to the data stored in a corresponding storage location.

In other words, Blumenau only teaches one to store state information to enable picking up exactly from where the error occurred. It thus teaches away from restarting, from the beginning, a write recovery process.

Claim 6 should be allowed.

There are additional reasons why at least Claim 6 should be allowed. The Examiner is of the opinion that the prior art further discloses that a load monitor process determines when a server is servicing a disproportionate share of client requests in a server group. The Examiner refers to Blumenau's paragraph [0064] for supposedly teaching this feature. There is a mention in that paragraph of monitoring when the storage system becomes "processor bound", "approaching its performance limit", or its "storage capacity". But these do not amount to a teaching, suggestion or even any inference that there is any determination of which share of client requests are being handled by a specific server, as claimed.

Claim 6 should therefore be allowed.

Claim 8 should be allowed.

Blumenau admittedly does disclose a state information table that is used to track the state of a storage element in a logical volume. For example, in paragraph [0051] is explained that state information 301 may include a count 302 which indicates a number of data operations performed on a corresponding storage element of volume 303. It is also stated in paragraph [0054] that state information may include other information used for recovery, such as to track information as needed for disk mirror processes.

However, there is no mention, suggestion, or teaching in Blumenau that this state table is used to perform any routing function among equivalent servers, or maintain data that would permit request routing based on the same.

Applicants' Claim 8, on the other hand, is directed to a routing table for tracking which resources are maintained on which servers in the system. Since Blumenau does not even disclose a plurality of storage servers that share a set of resources partitioned thereon, there is no need for him to either maintain such a routing table or determine how to route client requests among servers.

The Examiner also argues that paragraph [0064] in Blumenau teaches that a load monitor process monitors one or more of network traffic load, I/O request load or storage traffic pattern type. But the Examiner is merely repeating Applicants' claim language. None of these functions are actually stated in Blumenau, which merely determines when a storage system "becomes processor bound", "approaches a performance limit", or is "reaching storage capacity limits". These do not amount to teaching the more specifically claimed aspects monitoring of traffic load, I/O request load, or storage traffic pattern type.

Claim 8 is patentable over Blumenau.

Claims 14, 17 and 19 are also patentable.

Claim 14 is a method claim that recites:

> A process for moving resources across a storage system having a plurality of storage servers with a set of resources partitioned thereon, comprising …
> monitoring a load on a server…;
> transferring, as a function of the measured loads, a resource from one of said plurality of servers to another …
> detecting when a resource write request applies to a resource that is in the process of being resolved …; and
> in response to a write failure … restarting the migration process.

Claim 14 thus contains method steps that have corresponding features in claim 1. Claim 14 is thus allowable for the same reasons stated above for claim 1.

Claim 17 and 19 depend from claim 15 and recite method steps corresponding to features of claim 6 and 8, respectively. Claims 17 and 19 are thus patentable for the reasons stated above for claims 6 and 8.

Claim 5 is also patentable.

Claim 5 was rejected under 35 U.S.C. 103(a) as being obvious over Blumenau in view of Applicants' Admitted Prior Art (APA).

Claim 5 depends from Claim 1 and adds a requirement that the storage system is a Storage Area Network. Claim 5 is patentable for the same reasons as Claim 1 from which it depends.

Furthermore, there is no suggestion in the Blumenau or the Admitted Prior Art that a Storage Area Network can have a resource migration process that is restarted, from the beginning, upon detection of a migration write failure.

VIII. CONCLUSION

For the above stated reasons, all claims remaining in the application are patentable.

Respectfully submitted,

HAMILTON, BROOK, SMITH & REYNOLDS, P.C.

By /David J. Thibodeau, Jr., Reg. No. 31,671/

David J. Thibodeau, Jr.
Registration No.: 31,671
Telephone: (978) 341-0036
Facsimile: (978) 341-0136

Concord, MA 01742-9133
Dated: September 18, 2009

CLAIMS APPENDIX

1. (Rejected) An apparatus for resource migration, comprising a storage system having

a plurality of storage servers with a set of resources partitioned thereon, said storage servers having a load monitor process capable of communicating with other load monitor processes for generating a measure of loading on respective ones of the plurality of servers;

a resource migration process for transferring a resource from one of said plurality of servers to another of said plurality of servers in response to said measure of loading;

a write-detect process which:

(i) detects when a resource write request applies to a resource that is in the process of being moved from a first server to a second server, and which in response to such resource write request writes copies of the resource of both of said first and second server; and

(ii) in response to a write failure on the second server, restarts the migration process for the resource to ensure that the write request is propagated to the second server.

2. (Rejected) The apparatus of Claim 1, wherein said servers are equivalent to each other.

3. (Rejected) The apparatus of Claim 1, wherein said resources are selected from the group consisting of data blocks, program files, multimedia files, applications, and database files.

4. (Rejected) The apparatus of Claim 1, wherein said measure of loading reflects both a storage system load and a server load.

5. (Rejected) The apparatus of Claim 1, wherein said storage system is a Storage Area Network.

6. (Rejected) The apparatus of Claim 1, wherein the load monitor includes a process to determine whether a server is servicing a disproportionate share of the client requests being handled by a server group.

7. (Rejected) The apparatus of Claim 1, wherein the resource migration process includes a block data migration process.

8. (Rejected) The apparatus of Claim 1, further including a routing table for tracking resources maintained on the system.

9. (Rejected) The apparatus of Claim 1, wherein a pointer to a resource is maintained during an access operation to provide continuous data access.

10. (Rejected) The apparatus of Claim 1, wherein the load monitoring process monitors one or more of network traffic load, I/O request load, storage traffic pattern type.

11. (Canceled)

12. (Rejected) The apparatus of Claim 1, wherein the resource migration process divides the resource being moved into smaller subresources, and wherein the write-detect process;

(i) detects when a resource write request applies to a subresource that is in the process of being moved from a first server to a second server, and in response to such

resource write request writes copies of the subresource to both of said first and second server; and

(ii) wherein restarting the migration comprises restarting the migration for the subresource.

13. (Canceled)

14. (Rejected) A process for moving resources across a storage system having a plurality of storage servers with a set of resources partitioned thereon, comprising the steps of

monitoring a load on a server and communicating with other load monitor processes for generating a measure of loading on respective ones of the plurality of servers;

transferring, as a function of the measured loads, a resource from one of said plurality of servers to another of said plurality of servers in response to said measure of loading;

detecting when a resource write request applies to a resource that is in the process of being moved from a first server to a second server, and in response to such resource write request writing copies of the resource to both of said first and second server; and

in response to a write failure on the second server, restarting the migration process for the resource to ensure that the write request is propagated to the second server.

15. (Rejected) The process of Claim 14, wherein said servers are equivalent to each other.

16. (Rejected) The process of Claim 14, measuring a load includes measuring a storage system load and a server load.

17. (Rejected) The process of Claim 14, including the further step determining whether a server is servicing a disproportionate share of the client requests being handled by a server group.

18. (Rejected) The process of Claim 14, wherein the resource migration process includes a block data migration process.

19. (Rejected) The process of Claim 14, further including maintaining a routing table for tracking resources maintained on the system.

20. (Rejected) The process of Claim 14, wherein the load monitoring process monitors one or more of network traffic load, I/O request load, storage traffic pattern type.

21. (Rejected) The process of Claim 14, further including maintaining a pointer to a resource during an access operation to provide continuous data access.

22. (Canceled)

23. (Rejected) The process of Claim 14, further including:

dividing the resource being moved into smaller subresources;

detecting with the write-detect process when a resource write request applies to a subresource that is in the process of being moved from a first server to a second server, and in response to such resource write request writing copies of the subresource to both of said first and second server; and

wherein restarting the migration comprises restarting the migration for the sub resource.

24. (Canceled)

EVIDENCE APPENDIX

None.

RELATED PROCEEDINGS APPENDIX

None.